Screening Interacting Factors in a Wireless Network Testbed Using Locating Arrays

Randy Compton*, Michael T. Mehari†, Charles J. Colbourn*, Eli De Poorter†, Violet R. Syrotiuk*
* School of Computing, Informatics, and Decision Systems Engineering
Arizona State University, Tempe, AZ, USA 85287-8809
† Ghent University - iMinds
Department of Information Technology (INTEC)
Gaston Crommenlaan 8 (Bus 201), B-9050 Ghent, Belgium
{randy.compton,colbourn,syrotiuk}@asu.edu, {eli.depoorter,mmehari}@intec.ugent.be

***Abstract*—Wireless systems exhibit a wide range of configurable parameters (factors), each with a number of values (levels), that may influence performance. Exhaustively analyzing all factor interactions is typically not feasible in experimental systems due to the large design space. We propose a method for determining which factors play a significant role in wireless network performance with multiple performance metrics (response variables). Such screening can be used to reduce the set of factors in subsequent experimental testing, whether for modelling or optimization. Our method accounts for pairwise interactions between the factors when deciding significance, because interactions play a significant role in real-world systems. We utilize locating arrays to design the experiment because they guarantee that each pairwise interaction impacts a distinct set of tests. We formulate the analysis as a problem in compressive sensing that we solve using a variation of orthogonal matching pursuit, together with statistical methods to determine which factors are significant. We evaluate the method using data collected from the w-iLab.t Zwijnaarde wireless network testbed and construct a new experiment based on the first analysis to validate the results. We find that the analysis exhibits robustness to noise and to missing data.**

I. Introduction

An *experiment* is a series of tests in which purposeful changes are made to the parameters (factors) of a system in order to observe and identify the reasons for changes observed in an output response. Each *test* consists of a choice of value (level) for each factor. In general, experiments are used to study the performance of systems.

In this paper, the system studied is the w-iLab.t Zwijnaarde wireless network testbed [1]. The aim is to *screen* the measured responses of *mean opinion score* (MOS) [2] and the transmission exposure in a Wi-Fi audio broadcast, *i.e.*, to identify the factors and low order interactions that influence these two measured responses.

Methods for planning experiments seek to reduce the number of tests required to screen because the exhaustive *full-factorial design* [3], [4] is too large. For k factors each with two levels there are 2^k tests in the design. An *analysis of variance* (ANOVA) is readily calculated from such an experiment as it measures each factor and each t-way interaction, $2 \leq t \leq k$. From this, the important factors and interactions may be identified.

A *fractional factorial* design 2^{k-p} is a $\frac{1}{2^p}$ fraction of a full factorial design with k two-level factors. A fractional factorial design is *saturated* when it investigates $k = N-1$ factors in N tests [4]. In a *supersaturated design*, the number of factors $k > N-1$; these designs contain more factors than tests. However, a supersaturated design is not large enough to estimate all the factors, let alone any interactions [4]–[6].

A *D-optimal* design is one of the most popular experimental designs among those using optimality criteria. A model to fit, along with a bound on the number of tests, is specified a priori; this restricts the factors to be analyzed to those in the model.

Multi-objective optimizers have been proposed to optimize multiple, often competing, responses in wireless networks [7], [8]. The SUMO toolbox contains techniques for experiment generation, and the solution of such optimization problems. In [2], the toolbox was applied to a problem in a wireless network testbed using an exhaustive experiment. Necessarily, the number of factors was small. Our interest is not to eliminate factors from experimentation a priori. Instead, an automatic and objective approach to screening is sought.

Reducing the number of tests required therefore relies on a *sparsity of effects* assumption, that interactions of interest involve at most a small, known number t of interacting factors. As one means of reduction, *locating arrays* (LAs) have been defined [9]. For a set of factors each taking on a number of levels, an LA permits the identification of a small number of significant interactions among few (factor, level) combinations.

Locating arrays have been applied to screen factors and interactions in a simulated wireless network [10]. To the best of our knowledge, our work is the first to apply locating arrays for screening using data obtained from experimentation on a real system, a wireless network testbed.

A locating array is constructed to provide the screening experiment for a Wi-Fi audio broadcast with 24 factors having 2 to 5 levels each. An exhaustive experiment would contain over 23 trillion tests, but the locating array has only 109 tests. Each test is run on the testbed. A compressive sensing technique, orthogonal matching pursuit [11], is used to analyze the results. It recovers a preliminary model and then uses that model to decide which factors and levels, and which pairwise interactions between them, are important.

We then construct a second locating array using factors and levels deemed significant by the analysis of the first experiment. The resulting locating array for 8 factors has 94 tests but has higher coverage. The analysis for the second experiment differs both statistically and in terms of the constructed model from the first, so we repeated both experiments. The analyses of these later experiments resemble those of the second experiment. Indeed the later experiment with 109 tests indicates no significant factors that the first one missed, so the locating array with 94 tests remains appropriate. The analysis seems robust to high levels of noise, and also to holding constant the factors previously deemed insignificant.

II. Locating arrays

A *t-way interaction* is a set of t of the factors, and an admissible level for each. A *(d,t)-locating array* [9] on k factors is an $N \times k$ array such that, for every set of d distinct t-way interactions, the set of tests containing at least one of those interactions is not the same as the set of tests obtained from a different set of d distinct t-way interactions. Interactions may appear in different numbers of tests; an interaction has *higher coverage* if it appears in more tests.

Suppose the response measured in each test is pass or fail. A locating array permits locating the set of t-way interactions that cause the response of pass, provided there are at most d of them. A (d,t)-detecting array [9] is similar, but requires that the responses for a set of d t-way interactions rule out each t-way interaction not in the set of d. Reconstruction of the set of t-way interactions yielding a given set of responses is not known to be polynomial-time computable using locating arrays; when detecting arrays are used, efficient reconstruction is possible.

As we might expect, the responses measured in a real system, such as a wireless network testbed, have continuous values. However, significant interactions may not all have effects of the same magnitude. Therefore weak interactions may be indistinguishable from noise, so we seek to recover d or fewer *most significant* interactions above the noise floor. Locating arrays were not introduced to handle continuous responses, but even a $(1,2)$-locating array should suffice when the separation between effect magnitudes is large enough. Indeed the largest factor or interaction can be determined and its effect subtracted, leaving a similar problem for the remaining interactions. This suggests a "heavy hitters" approach for sparse model recovery.

III. Screening algorithm

A. Sparse model recovery

As in the field of compressive sensing, our objective is to determine a sparse representation of experimental data; in particular, we seek to minimize the ℓ_0 norm, the number of nonzero terms in the representation. Unfortunately minimizing using the ℓ_0 norm is NP-hard. Using the ℓ_2 norm permits solution via linear least squares regression, but it tends to produce representations with many more nonzero terms than necessary [12]. Nevertheless, there exist computationally tractable methods of better approximating the ℓ_0 norm, among them *orthogonal matching pursuit* (OMP) [11]. We utilize OMP because it iteratively identifies one term at a time to add to the representation.

Assuming sparsity of effects, because we screen for a small number of significant factors and low order interactions within a large number of possible ones, our problem can be cast in terms of sparse signal recovery. To turn a locating array into a compressive sensing matrix, we treat the factors as categorical and replace each test of the locating array with the 1-ary through t-ary combinations of the factor settings from that test, so that each way of setting the levels of factors involved in an interaction gets its own index and is thus available to be chosen separately in the sparse reconstruction. The resulting matrix provides multiple ways of representing some models.

B. Orthogonal matching pursuit

We modify orthogonal matching pursuit to solve our sparse recovery problem. Orthogonal matching pursuit incrementally builds a linear model of the data by choosing at each step the factor or interaction that best explains the residuals remaining from the previously-constructed model, then adds the chosen term to the model and recomputes the model coefficients. It uses the dot product of each candidate term with the residuals to select a term, because the term with the highest dot product is the closest to being orthogonal to the residuals. Then the chosen term is added to the model, the coefficients are recomputed via linear least-squares regression, and the residuals are recomputed as the difference between the original data and the new model's predictions [11].

We modify OMP to use Welch's unequal-variances t-test [13] instead of the dot product in term selection: We choose the term with the t-value of largest magnitude, because its explanation of some of the remaining variance is more statistically significant than that of any other term. If noise is (close to) normally-distributed, terms with higher coverage have lower expected variance; the variance is least when half the tests are selected by that term. Therefore the t-test automatically handles the effects of differing coverage. Because the model-building portion of OMP is unchanged, the residuals at each step are in a subspace orthogonal to all chosen terms, ensuring that the model explains as much of the data as possible for a linear model.

C. Significant factor identification

To avoid overfitting and to obtain a smaller model for screening, we sort the coefficients by decreasing absolute value, then take the largest coefficients prior to a cutoff. This cutoff is determined based on the R^2 value for the model built on just those coefficients — when it first crosses some pre-specified threshold or when its increase upon adding the next term to the model becomes sufficiently small. R^2 measures how much of the original variance remains after accounting for a model: $1-R^2$ equals the sum of squares of residuals divided by sum of squares of the original data after subtracting the mean, resulting in a range from 0 (no correlation) to 1 (perfect

reproduction of data points) [14]. Were the full computation too expensive, one could stop OMP early, but this could miss an important factor to be discovered later.

For screening, we retain factors appearing before the cutoff, whether alone or in an interaction with another factor, as significant. To determine levels to retain, we keep both extremes of each retained factor with an ordinal interpretation and each level chosen at least once before the cutoff for every type of factor. If only one level is not chosen, we also retain it, because the other levels must have been deemed significant in their contrast with it. For categorical factors, it is necessary to retain all levels due to lack of "extreme" value(s) for contrast. When there are multiple responses but a single set of factors is desired, we can consider the pre-cutoff terms from all responses together to determine which factors and levels may be significant to the whole system.

IV. Experimental Setup

A. The w-iLab.t testbed

Due to the increasing prevalence of network testbed facilities [15], [16], complex experimentation in wireless networks is now possible. The advantage of such facilities compared to simulation is that they correctly include the propagation and behaviour effects from the underlying physical layer and hardware. The iMinds w-iLab.t [1] at Zwijnaarde is one testbed that is used to perform heterogeneous wireless experimentation. The iMinds w-iLab.t testbed is pseudo-shielded from external interference and is equipped with various wireless technologies, including IEEE 802.11, IEEE 802.15.4, Bluetooth dongles, Software Defined Radios (SDRs), LTE femto cells, and others. The w-iLab.t testbed is part of Emulab and is controlled by the cOntrol Managment Framework (OMF) used for resource allocation, hardware and software configuration, and the orchestration of experiments. Finally, measurement data from each test is collected and stored in a central database over a wired control network for further processing.

B. Scenario

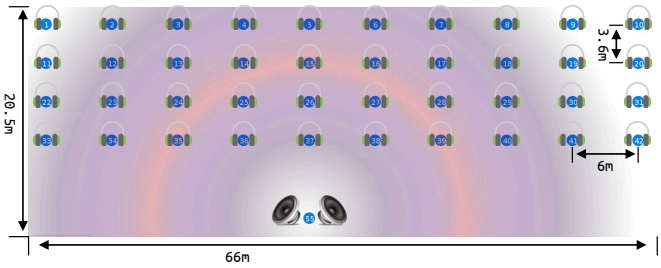

Fig. 1. High-level overview of the experiment scenario, as mapped to the wireless testbed. Listener nodes are in the first 4 rows (nodes 1-20, 22-31, and 33-42) and the speaker node is positioned at the bottom center (node 55).

A high-level representation of the Wi-Fi conferencing scenario created in the w-iLab.t testbed is shown in Fig. 1. It is composed of a speaker node broadcasting voice traffic over a Wi-Fi network and listener nodes receiving and playing the transmitted packets. The speaker can configure 24 different factors (described in §IV-C) that influence the transmissions. The listeners continuously calculate audio quality and transmission exposure.

The audio quality is quantified using an aggregate MOS [17] metric over the complete audio transmit path. During raw audio encoding at the speaker side, the transmission bitrate is reduced relative to the source material. The audio quality is expected to be further reduced when transmitted over the air because MOS is computed from packet loss, jitter, and latency.

Transmission exposure calculates the electromagnetic energy absorbed by a human body due to uplink and downlink wireless transmissions [2].

To orchestrate the experiment, an OMF script was written to process the experiment given by the locating array. The OMF script allocates resources and iteratively executes each test. During execution, the system is first brought to a known state by resetting all wireless interfaces and caches of each node, followed by configuration of the factors as specified by the test. After a warm-up period to avoid transient effects, measurements are collected. Table I shows the list of resources used for the Wi-Fi conferencing scenario.

TABLE I
Experiment Resource Description.

Resource	Description
Wi-Fi chipset	Atheros Sparklan WPEA-110N/E/11n
Wi-Fi driver	ath9k
OS	Ubuntu 14.04 LTS
kernel	Linux 3.13.0-33-generic
AP authenticator	hostpad 2.3
Client supplicant	wpa_supplicant 2.3

C. Selected factors and levels

Table II gives the factors and levels used in experimentation. We chose factors based on whether they could be set when a test was run and had potential relevance to our networking scenario, even if we expected an effect to be unlikely. (Recall that our goal is an automatic and objective approach to screening.) We selected 24 factors that may affect the Wi-Fi card, the IP and UDP stacks, and the audio codec. Except for the audio codec parameters, these were found from examination of the Ubuntu online documentation. In the end, sensing and ROHC were not implemented.

For the categorical (*e.g.*, qdisc) and ordinal factors, we selected up to 5 levels. We excluded the maximum Wi-Fi bitrate supported by both bands (54 Mbps) because the failure rate was too high to get useful data. For continuous factors other than percentages, we chose up to 5 exponentially-spaced levels to avoid privileging any particular scale, ensuring that the default (if any) was present while adding lower values under the assumption that the defaults are conservative. For percentages we chose 3 to 5 evenly spaced levels, except that if there is a default it is included.

TABLE II
FACTORS AND LEVELS USED IN EXPERIMENTATION (DEFAULT LEVELS FROM UBUNTU IN BOLD).

Factor	Identifier	Levels
Band	band	2.4, 5 GHz
Channel	channel	1, 6, 11 (2.4 GHz); 36, 40, 44 (5 GHz)
Wi-Fi bitrate	bitrate	6, 9, 12, 24, 36 Mbps
Transmit power	txpower	1, 2, 5, 10, **20** dBm
MTU	mtu	256, 512, 1024, 1280, **1500** bytes
Transmit queue length	txqueuelen	10, 50, 100, 500, **1000** packets
Queuing discipline	qdisc	pfifo, bfifo, **pfifo_fast**
IP fragment low threshold	ipfrag_low_thresh	25%, 50%, **75%**, 100% of high threshold
IP fragment high threshold	ipfrag_high_thresh	16384, 65536, 262144, 1048576, **4194304** bytes
UDP receive buffer minimum	udp_rmem_min	**1.9231%**, 10%, 50% of maximum
UDP receive buffer default	rmem_default	0%, 25%, 50%, 75%, **100%** from minimum to maximum
UDP receive buffer maximum	rmem_max	2304, 10418, 47105, **212992** bytes
UDP transmit buffer minimum	udp_wmem_min	**1.9231%**, 10%, 50% of maximum
UDP transmit buffer default	wmem_default	0%, 25%, 50%, 75%, **100%** from minimum to maximum
UDP transmit buffer maximum	wmem_max	4608, 16537, 59349, **212992** bytes
UDP global buffer minimum	udp_mem_min	25%, **50%**, 75% from minimum to maximum
UDP global buffer pressure	udp_mem_pressure	0%, **33.338%**, 50%, 75%, 100% from minimum to maximum
UDP global buffer maximum	udp_mem_max	95, 949, 9490, **94896** pages
Robust header compression	ROHC	off, on (unimplemented but present as a factor)
Audio codec	codec	Opus, Speex
Audio codec bitrate	codecBitrate	7600, 16800, 24000, 34000 bit/s (or nearest allowed by codec)
Frame length aggregation	frameLen	20, 40, 60
Interference channel occupancy	intCOR	10%, 25%, 50%, 75%, 90%
Sensing	sensing	off, on (unimplemented but present as a factor)

D. The locating array

The locating array constructed for our Wi-Fi scenario is a $(d = 1, t = 2)$-locating array, meaning it only guarantees to be able to locate (identify) at most one ($d = 1$) one-way or two-way (*i.e.*, up to $t = 2$-way) interaction in each iteration of the screening algorithm. It is remarkable that it distinguishes all one-way (*i.e.*, all (factor, level) combinations) *and* all two-way interactions (*i.e.*, all pairs of (factor, level) combinations) in only 109 tests.

V. ANALYSIS

A. Post-processing of measured responses

A complete set of responses was obtained for 25 different listeners in the experiment. The other 15 listeners failed to send any data to the controlling node for at least one test. These 15 were excluded from analysis, because sub-arrays of a locating array may not preserve the locating property, so a partial data set may fail to have an unambiguous interpretation. Determining values for the failed cases is problematic. We attempted to determine causes of the failures by treating each response as pass/fail and making use of the locating property of the array, but the failures appear not to be caused by a single level of a factor or 2-way interaction, which is all our array guarantees to locate. It is also possible that the failures could be intermittent, frustrating efforts to locate them.

We grouped the listening nodes' responses using hierarchical agglomerative clustering [18] to create an additional factor for location, which resulted in 3 levels of roughly equally-sized clusters. This factor was added because locations of the nodes across the testbed might have non-negligible effects on their performance due to metal objects, walls, other devices not participating in our experiment, and the like. The locating array remains $(1, 2)$-locating with the added factor.

Exposure measurements at the listeners yielded values much smaller than at the speaker, so we used just the exposure data from the speaker node, while MOS was available only at the listeners. Although our method could separately handle packet loss, jitter, and latency, we exclude them for brevity as their effects are already included in MOS. Computations for post-processing and analysis were performed using Octave [19].

To determine appropriate transformations to apply to the data, we produced probability plots for MOS and exposure. A straight line on a probability plot indicates the data fits the chosen probability distribution. For the first data set, MOS appears not to be uniformly or normally distributed, but it does appear closer to uniformly distributed after taking the exponential (see Fig. 2), so later analysis used the exponentials of the MOS values. Exposure, on the other hand, appears to be log-normally distributed (see Fig. 3), so the logarithms of the exposure values were used. As validation of our choice of transformation in the case of MOS, the total R^2 achieved by letting OMP execute until it the model has as many terms as there are runs is about 0.70 with the exponential transformation, while the untransformed data yields a final R^2 of approximately 0.35. This suggests that the exponential transformation allows the model to account for about twice as much of the variance in the data.

B. Results and confirmation experiments

The top factors and interactions for both responses for the first data set are given in Tables III and IV, along with the R^2 value for the model so far. Only the signs of the terms are shown because their model coefficients change as more terms are added to the model. The OMP rank denotes the order in

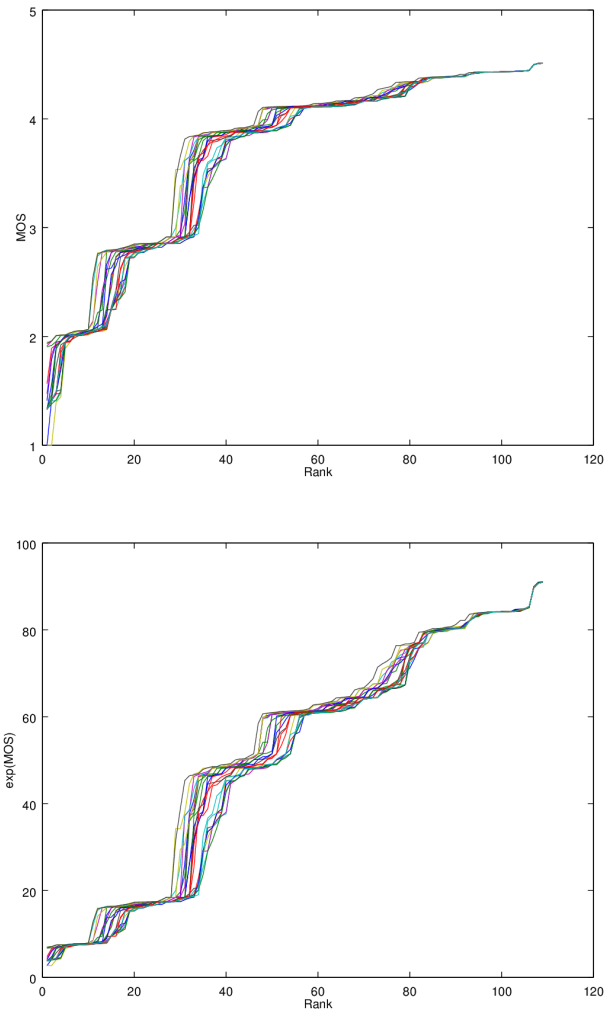

Fig. 2. Uniform (top) and exponential-uniform (bottom) distribution plots for MOS for all listener nodes that reported measurements in each test.

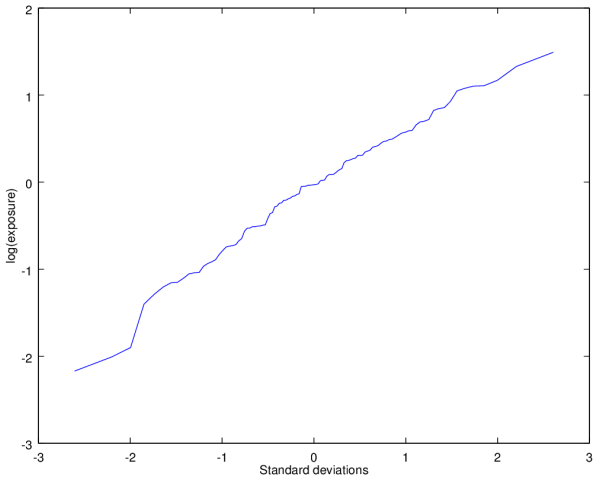

Fig. 3. Log-normal distribution plot for exposure at speaker node.

which the terms were selected by OMP prior to being sorted by coefficient. The cutoffs are shown when R^2 changes by a negligible amount, here taken to be less than 0.01.

TABLE III
TOP TERMS IN EXP(MOS) MODEL; CUTOFF AFTER SECOND ROW.

Rank	OMP Rank	Sign	Factor(s)	R^2
1	1	−	codecBitrate=7600/7750	0.474
2	2	−	codecBitrate=16800	0.551
3	15	+	codec=opus×codecBitrate=16800	0.555
4	11	−	channel=11/44	0.561
5	18	−	ipfrag_low_thresh=1×udp_rmem_min=0.1	0.571
6	37	+	channel=11/44×udp_mem_min=0.75	0.571
7	5	−	intCOR=0.9	0.592
8	34	+	sensing=1	0.592
9	23	−	channel=6/40×codec=speex	0.596
10	27	+	udp_rmem_min=0.1×wmem_max=59349	0.596

TABLE IV
TOP TERMS IN LOG(EXPOSURE) MODEL; CUTOFF AFTER SIXTH ROW.

Rank	OMP Rank	Sign	Factor(s)	R^2
1	1	−	rate=36	0.350
2	3	+	frameLen=20	0.432
3	4	+	band=5×channel=11/44	0.545
4	6	+	rate=6	0.695
5	2	−	rate=24	0.838
6	5	+	codecBitrate=34000/34200	0.882
7	10	+	frameLen=40	0.884
8	20	−	band=2.4×intCOR=0.5	0.886
9	11	+	rate=9	0.895
10	8	+	band=2.4×channel=1/36	0.906

In order to confirm the screening results, a follow-up experiment is conducted. For this experiment, we generated another locating array on the 8 significant factors screened by the first locating array: band, channel, Wi-Fi bitrate, codec, codec bitrate, frame length aggregation, and interference channel occupancy. The last factor has just the 10%, 50%, and 90% levels retained and the others have all their original levels. All other factors are held constant at their defaults. This locating array is a $(2, 2)$-locating array with 94 tests; we increased the locating ability because we have fewer factors.

The experiment using this locating array resulted in a second data set, in which 28 of the 35 available listener nodes reported measurements in all tests. The MOS distribution for this experiment has similar shape to the first experiment. The exposure measurements appear log-uniformly rather than log-normally distributed and also cover a different range. The cause of this difference is unknown, so we re-ran both experiments again, for which 36 listener nodes were available. Of these 27 reported in all tests of the experiment with 109 tests and 28 reported in all tests of the experiment with 94 tests. The data set from the repeated experiment with 109 tests has similar MOS distributions to the others, and the exposure measurements resemble the second data set, so it appears that whatever changed between the first and second experiments remained the same thereafter.

The data set from the repeated experiment with 109 tests agrees with the second experiment. We find that it selected

a subset of the factors and levels that the first experiment selected, so the tests used in the second experiment are still justified. For MOS for all data sets, the location factor we added did not appear significant either in isolation or in interactions, although including it did change the full model and its R^2 value slightly.

For further justification of our analysis, we compared the data sets pairwise using an additional "set" factor to denote which set each data point came from. The intuition is that, if the "set" factor is considered significant, we have reason to suspect it has some model-relevant difference. We also used the Kolmogorov-Smirnov test [20] to compare the statistical distributions of the data sets for reference. We find perfect agreement between whether the "set" factor was selected before $R^2 \geq 0.9$ and whether the Kolmogorov-Smirnov test would reject the distributions' equality at the 0.05 significance level, providing a consistency check of our analysis. Although the MOS distributions of both 109 test experiments differ from each other and from both 94 test experiments, the data sets all select just the codec bitrate as significant, except the second data set that also selects codec type. This suggests that our method has some robustness to noise.

We also analyzed each data set including the nodes with missing data points, adding a factor to distinguish them from those with all values present. Each found no significance in whether all data points were present, so our choice to exclude data from incomplete nodes does not affect the conclusions.

VI. Conclusions and Future Work

To the best of our knowledge, this paper is the first to present results of screening using locating arrays in a physical system that is likely to have non-negligible interactions. A simulation might lack interactions that appear in real systems or include spurious interactions due to aspects of the hardware that are not well-modelled. Indeed the method could be used to compare the results of experimentation in a simulated to a physical wireless network.

Our analysis method applied to the results of tests selected by a locating array is able to rule out most insignificant factors and levels, which can in turn significantly reduce the number of runs required in later experimentation. Responses are relatively consistent across experiments using different locating arrays, suggesting its validity even in the absence of a formal proof of locating arrays' sufficiency for continuous-valued outcomes. Future work may verify this or find another design that combines the locating property with the high-confidence reconstruction guarantees of compressive sensing.

In some systems there may be factors that contribute significantly to the outcome but cannot be controlled, such as ambient temperature. More work is necessary to determine how to detect and handle uncontrollable factors, and in cases when they cannot even be observed, to detect their existence and influence indirectly.

Screening is just the first step in model building and optimization. Determining the levels of the factors screened by our method that maximize MOS while minimizing transmission exposure is the next step in experimentation. Validation by other methods, such as those used in SUMO [2], is of interest.

VII. Acknowledgements

We thank Ingrid Moerman for her support. This work is supported in part by the National Science Foundation under Grant No. 142105, by the IWT project "SAMURAI: Software Architecture and Modules for Unified RAdIo control," and by the European Commission Horizon 2020 Programme under grant agreement n 688116 (eWINE).

References

[1] S. Bouckaert, W. Vandenberghe, B. Jooris, I. Moerman, and P. Demeester, "The w-ilab.t testbed," in *Testbeds and Research Infrastructures. Development of Networks and Communities*, T. Magedanz, A. Gavras, N. H. Thanh, and J. S. Chase, Eds. Springer Berlin Heidelberg, 2011, pp. 145–154.

[2] M. T. Mehari, E. De Poorter, I. Couckuyt, D. Deschrijver, J. V.-V. Gerwen, D. Pareit, T. Dhaene, and I. Moerman, "Efficient global optimization of multi-parameter network problems on wireless testbeds," *Ad Hoc Networks*, vol. 29, pp. 15–31, 2015.

[3] C. Croarkin, P. Tobias, J. J. Filliben, B. Hembree, W. Guthrie, L. Trutna, and J. Prins, Eds., *NIST/SEMATECH e-Handbook of Statistical Methods*. NIST/SEMATECH, April 2012.

[4] D. C. Montgomery, *Design and Analysis of Experiments*, 7th ed. John Wiley and Sons, Inc., 2009.

[5] S. Gilmour, "Factor screening via supersaturated designs," in *Screening*, A. Dean and S. Lewis, Eds. Springer New York, 2006, pp. 169–190.

[6] R. Li and D. K. J. Lin, "Analysis methods for supersaturated designs: Some comparisons," *Journal of Data Science*, pp. 249–260, 2003.

[7] K. Navaie and T. A. Le, "Fundamental performance trade-offs in coexisting wireless networks," in *5G for Ubiquitous Connectivity (5GU), 2014 1st International Conference on*, November 2014, pp. 246–251.

[8] R. Annauth and H. C. Rughooputh, "OFDM systems resource allocation using multi-objective particle swarm optimization," *International Journal of Computer Networks and Communications*, vol. 4, no. 4, pp. 291–306, July 2012.

[9] C. J. Colbourn and D. W. McClary, "Locating and detecting arrays for interaction faults." *J. Comb. Optim.*, vol. 15, no. 1, pp. 17–48, 2008.

[10] A. N. Aldaco, C. J. Colbourn, and V. R. Syrotiuk, "Locating arrays: A new experimental design for screening complex engineered systems," *SIGOPS Operating Systems Review*, vol. 49, no. 1, pp. 31–40, January 2015.

[11] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, December 2007.

[12] E. Candès and M. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[13] B. L. Welch, "The generalization of 'student's' problem when several different population variances are involved," *Biometrika*, vol. 34, no. 1/2, pp. 28–35, 1947.

[14] N. R. Draper and H. Smith, *Applied Regression Analysis (3rd Edition)*. Somerset, NJ, USA: Wiley, 2014.

[15] "GENI: Global Environment for Network Innovations," https://www.geni.net/.

[16] "FIRE: Future Internet Research & Experimentation," http://www.ict-fire.eu/home.html.

[17] M. T. Mehari, E. De Poorter, I. Couckuyt, D. Deschrijver, G. Vermeeren, D. Plets, W. Joseph, L. Martens, T. Dhaene, and I. Moerman, "Efficient identification of a multi-objective pareto front on a wireless experimentation facility," *IEEE Transactions on Wireless Communications*, under review.

[18] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[19] J. W. Eaton, D. Bateman, S. Hauberg, and R. Wehbring, *GNU Octave version 3.8.1 manual: a high-level interactive language for numerical computations*. CreateSpace Independent Publishing Platform, 2014.

[20] I. M. Chakravarti, R. G. Laha, and J. Roy, *Handbook of Methods of Applied Statistics*. John Wiley and Sons, 1967, vol. I.